A Model for Autonomous Reconfiguration of Sensing Web Resources

Robert Morris
Jennifer Dungan
Lina Khatib
Petr Votava

NASA Ames Research Center

Topics covered in this talk

- Technical objective: reconfiguring sensor webs
- Approach based on automated planning and execution
  - Hierarchical, model-based workflow generation and execution
- System overview
- Component description
- Status and future work

Sensor Web Configuration

- An Earth Science *Sensor Web* is a distributed, coordinated system of sensors, models, humans, data sets and data repositories.
- Humans *configure* sensor webs to answer science questions, or for disaster management.
  - Webs are (re)configured through a series of commands or requests.
- Web configuration can be time-consuming and difficult for humans to perform.
  - Requires managing a complex, distributed system.
- Automation can offer benefits in the process.

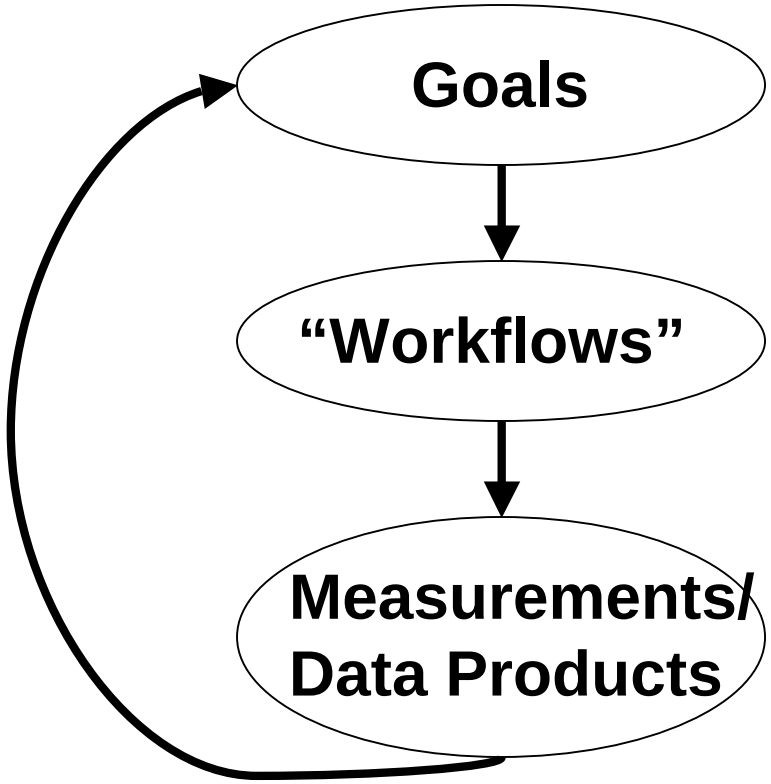
Requirements for Automation

- Improve accessibility to web resources
  - Coordination of resources
- Demonstrate improvements in quality of data
- Generality:
  - Improve the process of data acquisition for any Earth science focus area
- Robustness:
  - Adapt to changes in resource availability during the (re)configuration process.
- Abstract acquisition details from user
  - Define goals at a high level of abstraction

Summary of Approach

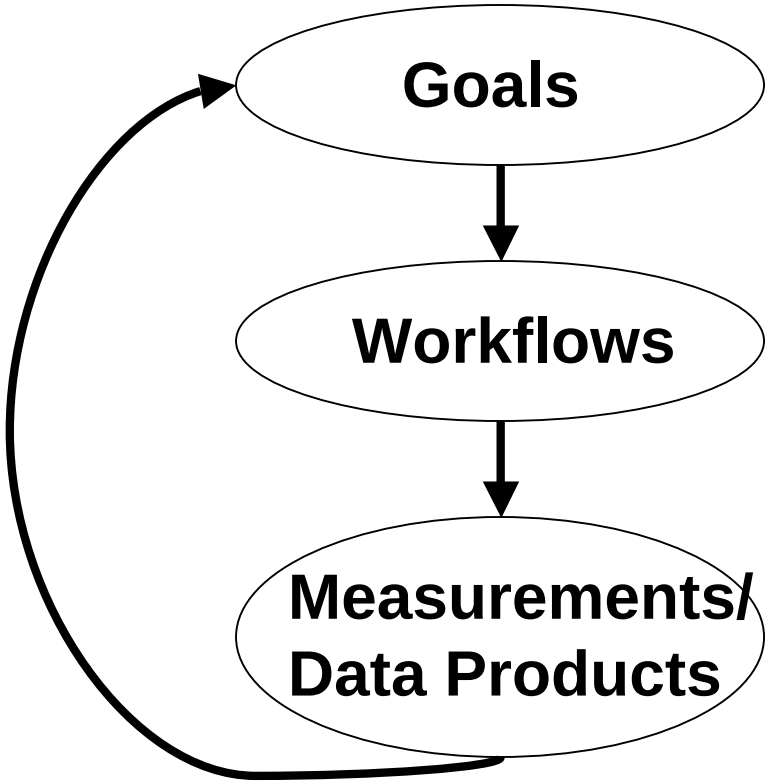
- XML-based goal specification
  - Specifies constraints on data product, workflow
- Automated model-based workflow generation
  - Process based language for composing workflows
- Automated workflow execution
  - Hierarchical model of workflow (abstract, concrete)
  - Executable workflow as a finite state machine
- Leveraging OGC/SWE service layer
- Testbed for demonstrating ideas using Terrestrial Observation and Prediction System (TOPS)

The Data Acquisition Cycle



Workflows are plans for reconfiguring a sensor web to achieve goals.

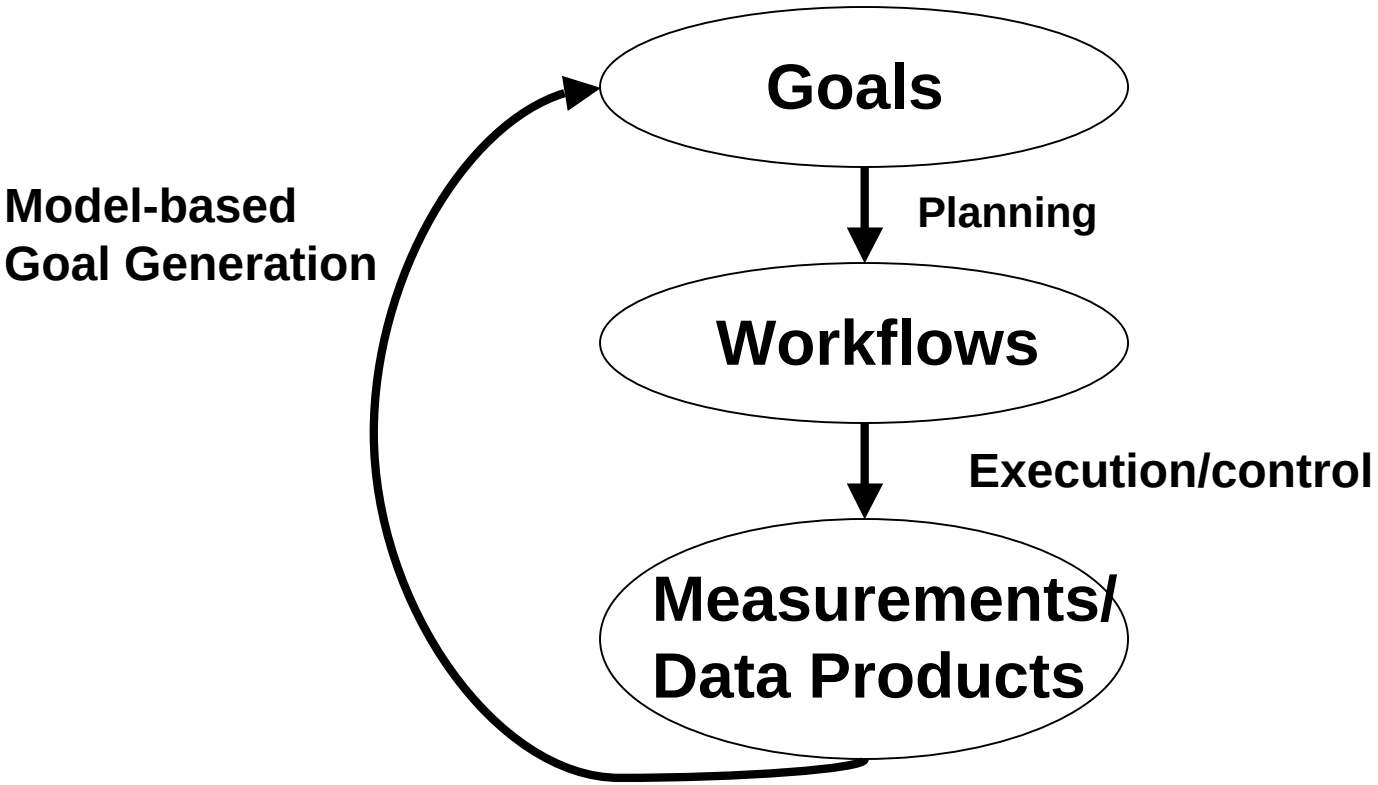
Example



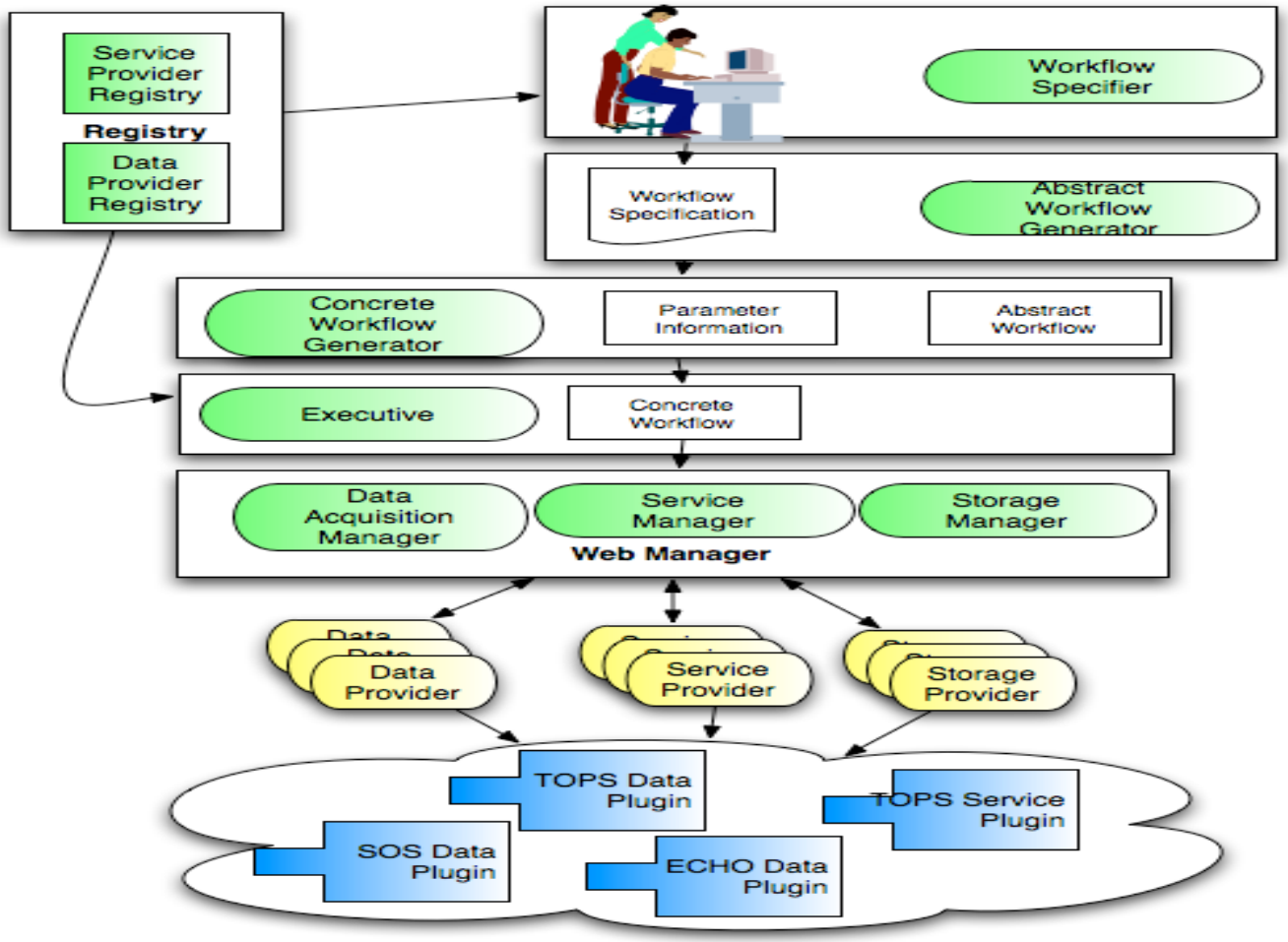
Goal: ***Characterize*** *vegetation for a specified region during 2006 using MODIS data.*

Workflow: ***Get*** *MODIS leaf area index (lai) data between 01/01/2006 and 12/31/2006 in a region defined by a bounding box defined by lower corner lat-long <25.00, -110.00> and upper corner lat-long <40.00, -125.00> and* ***process*** *them every month using the average.* ***Store*** *the measurements into hdf files on the FTP site.*

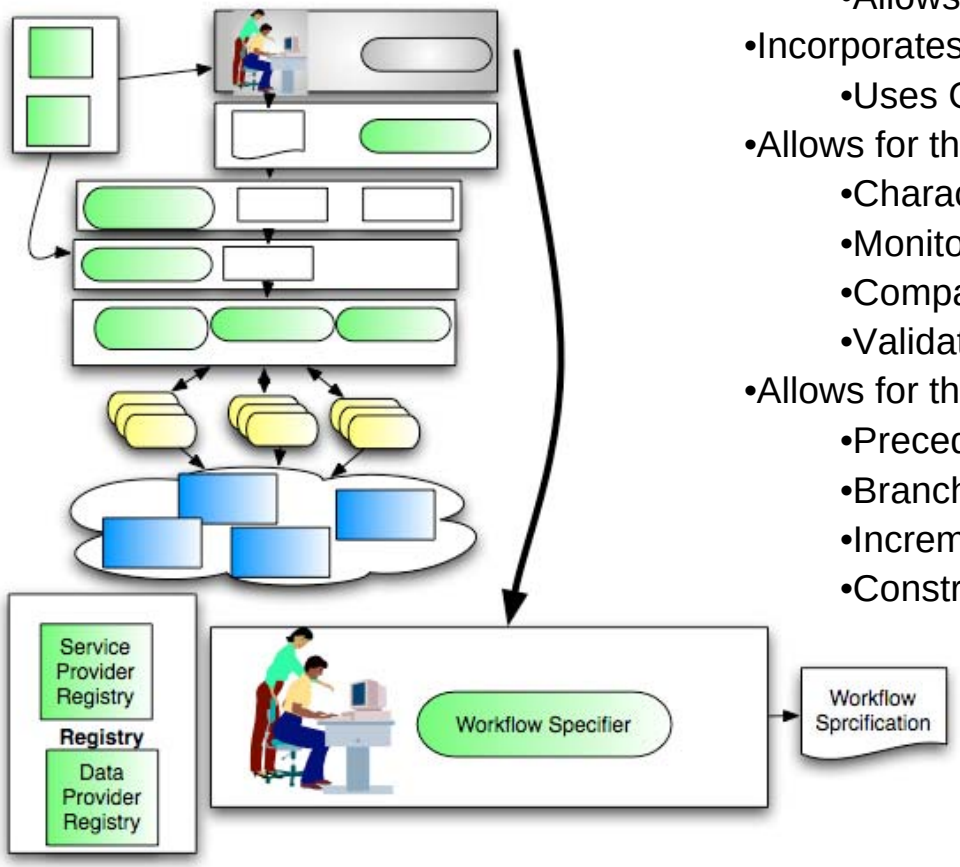
Relevant Software Technologies



Layered Architecture



Workflow Specification

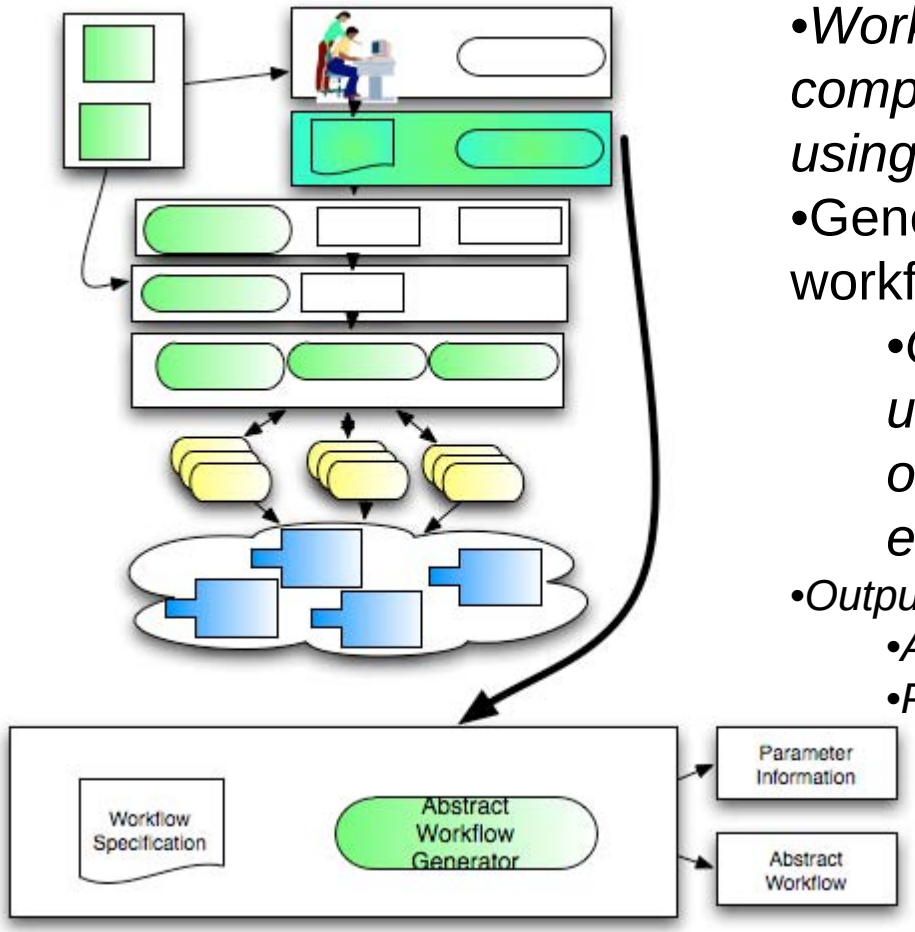


- Goals are expressed using a markup language (GoalML)
  - Facilitates information description in a nested structure
  - Allows tools for validation and parsing to be used
- Incorporates and extends Geography Markup Language (GML)
  - Uses GML spatial and temporal model
- Allows for the definition of workflows for different science goals
  - Characterizing/classifying/Mapping
  - Monitoring an event or measurements over time
  - Comparing/Conflating data sets
  - Validating sensors
- Allows for the definition of constraints on the workflow
  - Precedence and other temporal constraints
  - Branching on results of actions, events in the world
  - Incremental retrieval (staged processing)
  - Constraints on data quality

Goal Taxonomy

- ***Compare*** *the current year's annual net primary production (NPP) map for North America with the 25-year long-term average NPP.*
- ***Monitor*** *daily sea surface temperature for the Gulf of Mexico for 2000-2006*
- ***Map*** *land cover types in 2001 for the state of Sonora, Mexico using the International Geosphere-Biosphere Programme (IGBP) global vegetation classification scheme from the MOD12Q1 product.*
- ***Predict*** *monthly NPP for North America in 2010 using the prognostic BGC model in TOPS.*
- ***Validate*** *the GLOBCARBON leaf area index product for June, 2005, with measurements made at Fluxnet Tower loctions in Canada.*
- ***Compare*** *MOZART model predictions of CO concentrations in an area around Fairbanks Alaska with Observations taken by AIRS. If the predictions diverge from the observations by more than a certain amount,* ***characterize*** *the CO concentrations for the same area for tomorrow. (Composite goal)*

Workflow Generation



- *Workflows are **finite state processes** composed from a small set of basic **tasks** using a small set of **operators**.*
- Generation Algorithm: Enables automatic workflow composition
  - *Composes workflows from **goals** using a model consisting of rules for composing workflow elements*
- *Output of generator consists of*
  - *Abstract plan (workflow constraints)*
  - *Parameter information (constraints on product)*

Workflow Generation from Goals

- Hierarchical workflow representation
- *Abstract workflow:* defines workflow structure without referring to specific resources for task execution.
  - Provides flexible way to define workflows without being concerned with low-level details.
  - Tasks are portable and can be mapped at run time to suitable web services
- *Concrete workflow:* a binding of tasks to resources
    - Tasks move data around or deliver data products to user

Composing Workflows from Goals using a Process Model

- An abstract workflow is described in terms of a *Finite State Process (FSP)*
  - An FSP describes the execution of a sequence of basic tasks
  - Two kinds of basic tasks: data acquisition and "staging"
    - Data acquisition tasks include *get, process (data), store.*
    - Staging tasks allow for the workflows to be autonomously executed and monitored.
      - Example: test whether a server is available
      - Test whether data is ready to be acquired.

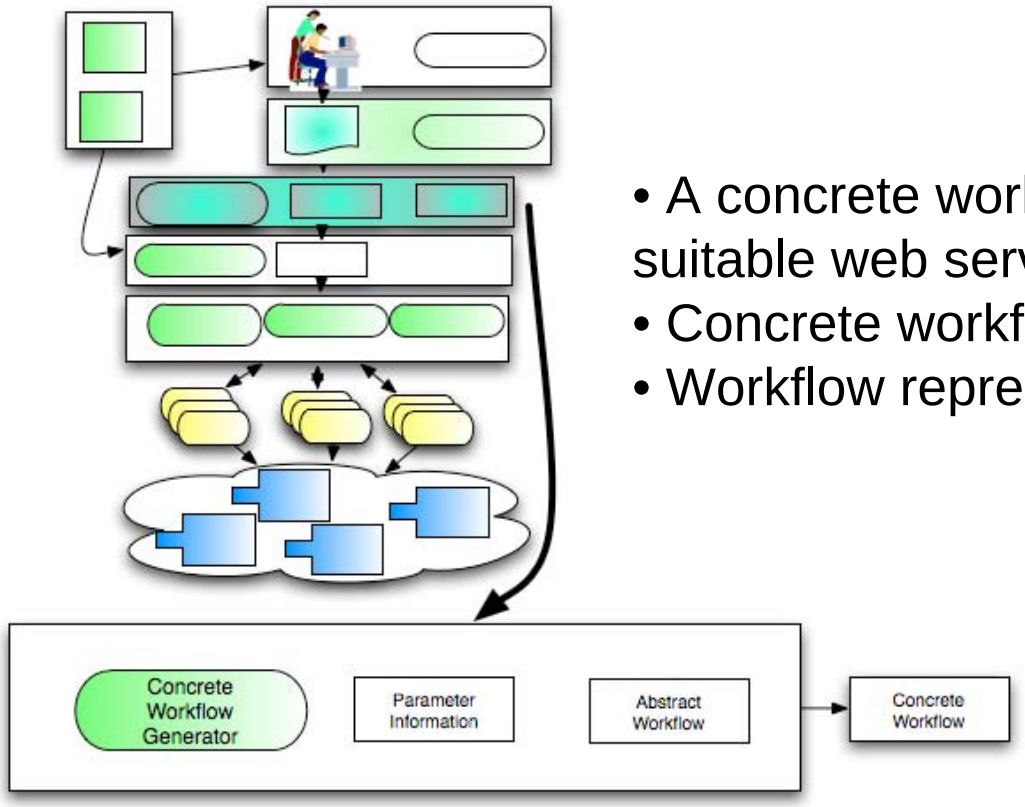
Composing Workflows from Goals using a Process Model

- Operators for composing workflows: If *P,Q* are processes and *a,b* are basic actions, then
  - **Sequence:** *a --> P* is a process
  - **Choice:** *(a --> P | b --> Q)* is a process
  - **Recursion:** *P: a --> P* is a process
  - **Non-determinism:** *(a --> P | a --> Q) is a process*
  - **Parallelism:** *(P || Q)* is a process.
  - **Instance/type:** P1:P P1 is a process of type P
- Implementation using Labeled Transition State Analyzer (LTSA) system
  - Verification tool for concurrent systems

Simple Abstract Workflow

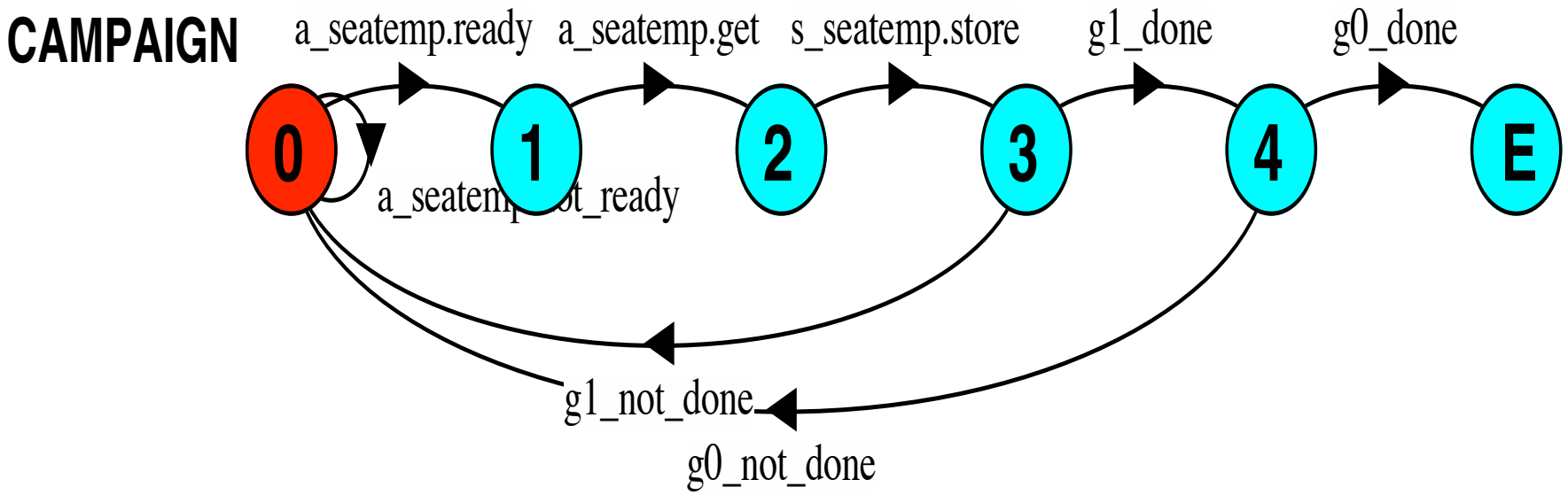
```
//Declaration section
ACQUISITION = (ready -> get -> END | not_ready -> ACQUISITION).
PROCESS     = (process -> END).
STORE       = (store -> END).
TEST        = (test -> (test.true->END | test.false->END)).

//Goal#0
                //Goal#1
                 //components:
                    ||A_SEATEMP = (a_seatemp:ACQUISITION).
                   ||S_SEATEMP = (s_seatemp:STORE).
                 //workflow:
                      G1_WFC1          = A_SEATEMP;S_SEATEMP;END.
                      ||G1_ONCE        = (A_SEATEMP||S_SEATEMP||G1_WFC1).
                      G1_REPEAT        = G1_ONCE;G1_TEST4DONE,
                      G1_TEST4DONE     = (g1_done -> END | g1_not_done -> G1_REPEAT).
                      ||G1             = (G1_REPEAT).
 //Workflow:
      ||G0_ONCE     = (G1). //Parallel composition of components and workflow constraints
      G0_REPEAT     = G0_ONCE;G0_TEST4DONE,
      G0_TEST4DONE = (g0_done -> END | g0_not_done -> G0_REPEAT).
      ||G0          = (G0_REPEAT).

||CAMPAIGN = (G0).
```

Concrete Workflow Generation



- A concrete workflow binds acquisition actions to suitable web services at run time.
- Concrete workflows are *executable*.
- Workflow represented as a finite state machine.

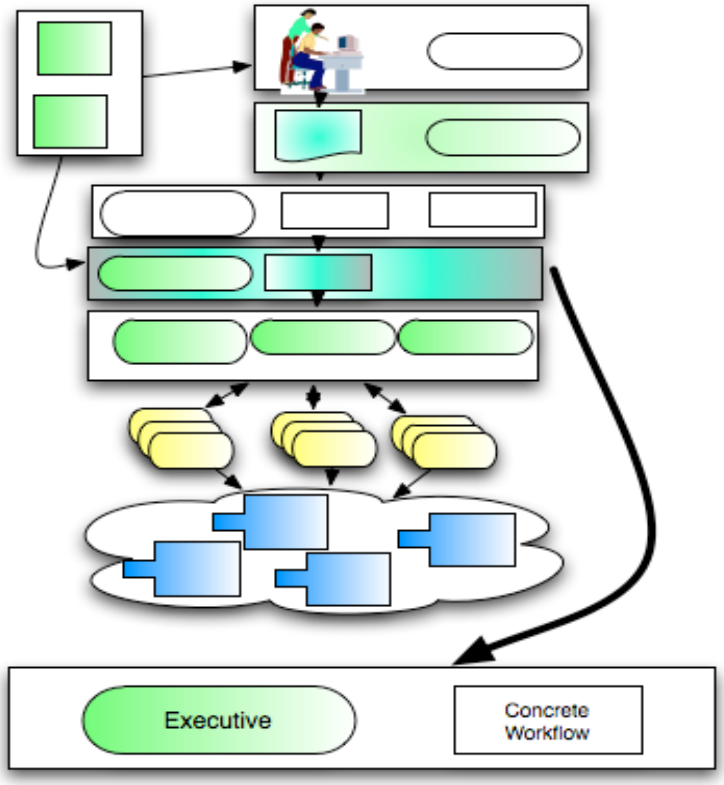
Workflow as a Finite State Machine



Concrete Workflow Generation

- Combines abstract Workflow in the form of finite state machine with concrete instantiation of the GoalML document
- Utilizes schema that describes the structure of the Goal language
- Translation of the schema to java implementation using the Java Architecture for XML Binding (JAXB)
- In addition, integrates the following from OGC templates:
  - OGC Geographic Markup Language (GML)
  - OGC Sensor Web Enablement Common Data Model (SWE-Common)
- Currently over 220,000 lines of code and more than 1100 java classes
  - Ability to process and validate *any* GoalML, GML and SWE-Common documents
  - Used the experience to derive bindings for SensorML
  - Will be also useful for Sensor Observation Services and other standards

Workflow Execution



- Executes a workflow as a finite state process.
- Monitors progress of execution, enables tolerance to execution failure (robustness)

Workflow Execution

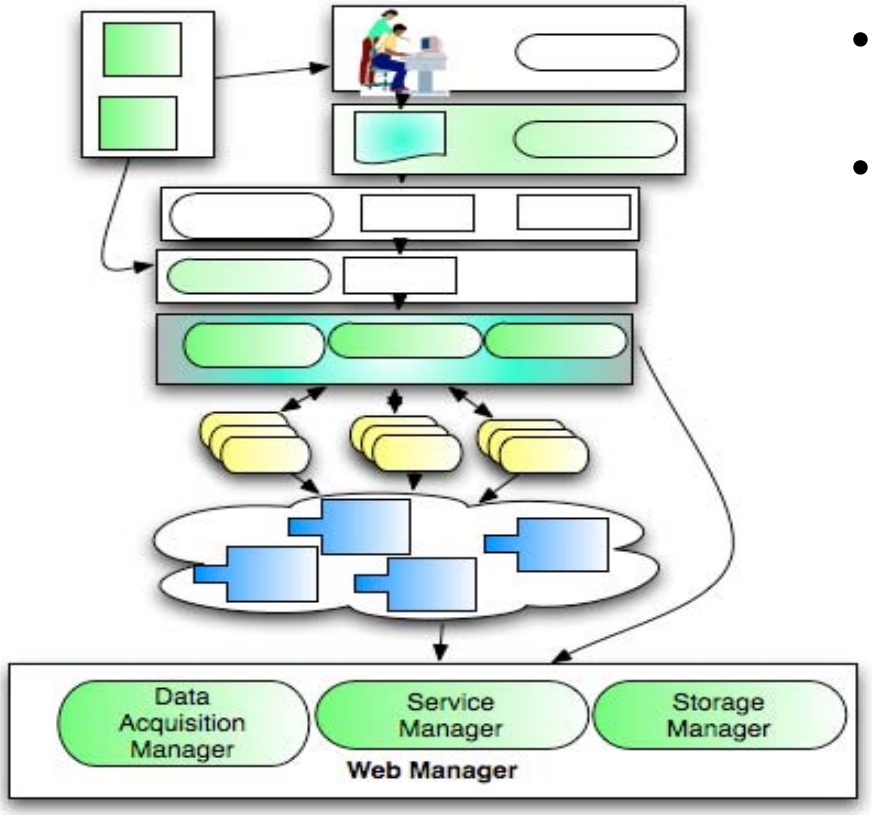
Algorithm:

- Incremental composition of **concrete workflow** out of **abstract workflow** and **procedure mapping** function.
- Concrete workflows are executed by incrementally mapping basic tasks in an abstract plan to requests for web services.
- Observation of progress towards goals.

Representation:

- Finite state machine-based representation of workflows
- Coordination with Web Manager to perform mapping

Web Manager



- High level interface to web service layer
- Consists of three main components:
  - Data Manager
    - Provides access to the sensor web data
  - Service Manager
    - Provides access to services (processes) - local or remote
  - Storage Manager
    - Provides access to storage (filesystem, database)

Data Management

- *Data Manager* manages a collection of *Data Providers*
- *Data Provider* consists of an entry point (e.g. location on the web, such as address and a port) and *Data Service*
- *Data Service* is an interface that provides generic view of any data service
  - Users and developers can implement any service they want/need as well as reuse existing *Data Service* implementations
  - TOPS, ECHO, Sensor Observation Services, …
  - Flexible architecture, so they can be loaded in runtime without any change to the system code or the need to re-compile
  - Advantage when designing architecture for unknown future services

Process Management

- Similar flexible architecture with plug-in services
- Ability to handle both local and remote services,
  - Works across styles/protocols such as REST, SOAP, CORBA, etc. - the particular plug-in will handle the low level details
- Currently using SensorML process model to describe local processes and the constraints on their inputs and outputs
  - May not be always available and we would like the ability to interface other systems without having to force SensorML on the users

Integration into TOPS

Ames Research Center

ECOCAST

Monitoring, Modeling, and Forecasting Ecosystem Change

Home | Images | Data | TOPS | Projects | Applications | Education | Pubs | People

NASA Ames Ecological Forecasting Lab

Welcome to the Ecological Forecasting Lab at NASA Ames Research Center. Our mission is to use the Terrestrial Observation and Prediction System (TOPS) to develop nowcasts and forecasts of ecosystem conditions for use in a range of applications.

TOPS is a data and modeling software system designed to seamlessly integrate data from satellite, aircraft, and ground sensors with weather/climate and application models to produce operational nowcasts and forecasts. TOPS operates at a variety of spatial scales, ranging from individual vineyard blocks to global monthly assessments of vegetation net primary production.

MODIS Aqua Image of San Francisco Bay, 18-Jun-2008

browse images & data

View Images

View daily updates of biospheric parameters. See below for a selection of available regions or view images here.

Regions

- Overview
- Global
- Continental U.S.
- California
- Yosemite

NASA Official: Ramakrishna R. Nemani
Curator: Forrest Melton
Privacy Statement

Last Updated: Jan 3, 2007

Status and Future Directions

- Integration and testing
  - Demonstrating end-to-end capabilities on numerous use cases
  - Using TOPS as initial test bed.
- Distributed planning and execution
  - Leverage modularity of process-based approach
- Managing uncertainty in execution
- Deployment in distributed applications

Summary

- Developed a model of reconfiguring sensor webs using principles from autonomous workflow planning and execution
- Emphasis on
  - generality of approach
  - Abstraction for greater flexibility
- Model-based workflow generation
- Framework for autonomous execution of workflows in conjunction with service layer